



UPPSALA  
UNIVERSITET

TVE 14 056 augusti

Examensarbete 15 hp  
Augusti 2014

# Inertial sensor and ultra-wideband sensor fusion

Precision positioning of robot platform

---

Mikael Frisk  
Albin Nilsson



UPPSALA  
UNIVERSITET

**Teknisk- naturvetenskaplig fakultet  
UTH-enheten**

Besöksadress:  
Ångströmlaboratoriet  
Lägerhyddsvägen 1  
Hus 4, Plan 0

Postadress:  
Box 536  
751 21 Uppsala

Telefon:  
018 – 471 30 03

Telefax:  
018 – 471 30 00

Hemsida:  
<http://www.teknat.uu.se/student>

## Abstract

### Inertial sensor and ultra-wideband sensor fusion

---

*Mikael Frisk Albin Nilsson*

In order to determine the orientation and position of an object it is common to measure linear and angular motion with an inertial sensor. To further improve the positioning an ultra-wideband sensor can be used simultaneously and integrated in the final solution with sensor fusion. This study has evaluated an ultra-wideband sensor, and also integrated it with a pre-existing solution for positioning using inertial sensors, in order to determine if the solution is viable for positioning of an autonomous robot platform for use in industrial environments. The results of the evaluation shows that the ultra- wideband sensor would work well in the intended environments but that further tests are needed to establish behavior in none-line-of-sight conditions. The results from the sensor fusion show that the integration of the two sensors yields a result that is better than either of the sensors working alone. The main issue with the current solution is a low sampling rate of the ultra-wideband sensor. A list of other suggested improvements is provided.

Handledare: Thomas Schön  
Ämnesgranskare: Petter Tammela  
Examinator: Martin Sjödin  
ISSN: 1401-5757, TVE 14 056 augusti

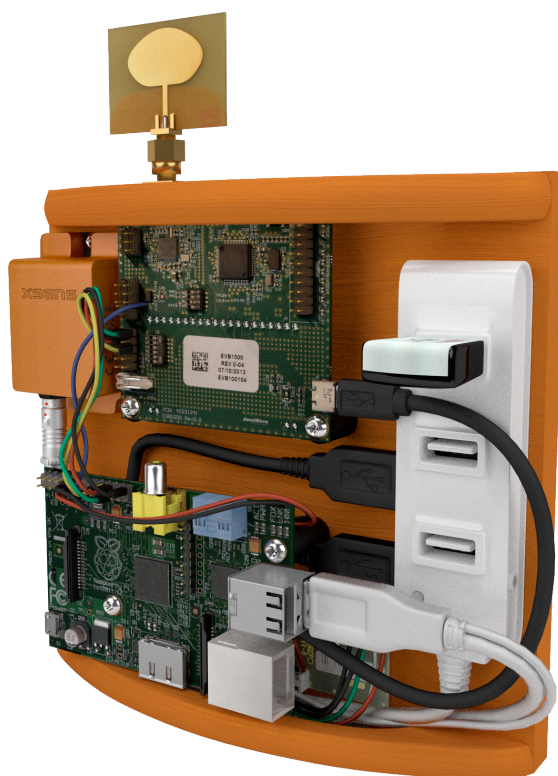


**xsens**

# Inertial sensor and ultra-wideband sensor fusion

## Precision positioning of robot platform

**Mikael Frisk**  
**Albin Nilsson**



Xsens Technologies B.V.

Pantheon 6a  
P.O. Box 559  
7500 AN Enschede  
The Netherlands

**phone** +31 (0)88 973 67 00  
**fax** +31 (0)88 973 67 01  
**e-mail** [info@xsens.com](mailto:info@xsens.com)  
**internet** [www.xsens.com](http://www.xsens.com)

Xsens North America, Inc.

10557 Jefferson Blvd,  
Suite C  
CA-90232 Culver City  
USA

**phone** 310-481-1800  
**fax** 310-416-9044  
**e-mail** [info@xsens.com](mailto:info@xsens.com)  
**internet** [www.xsens.com](http://www.xsens.com)



Abstract: In order to determine the orientation and position of an object it is common to measure linear and angular motion with an inertial sensor. To further improve the positioning an ultra-wideband sensor can be used simultaneously and integrated in the final solution with sensor fusion. This study has evaluated an ultra-wideband sensor, and also integrated it with a pre-existing solution for positioning using inertial sensors, in order to determine if the solution is viable for positioning of an autonomous robot platform for use in industrial environments. The results of the evaluation shows that the ultra-wideband sensor would work well in the intended environments but that further tests are needed to establish behavior in none-line-of-sight conditions. The results from the sensor fusion show that the integration of the two sensors yields a result that is better than either of the sensors working alone. The main issue with the current solution is a low sampling rate of the ultra-wideband sensor. A list of other suggested improvements is provided.



# Table of Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>1</b>
1.1	BACKGROUND .....	1
1.2	PURPOSE .....	1
1.2.1	<i>The robot platform .....</i>	<i>1</i>
<b>2</b>	<b>THEORY .....</b>	<b>3</b>
2.1	IMU SENSORS .....	3
2.1.1	<i>IMU accelerometers .....</i>	<i>3</i>
2.1.2	<i>IMU gyroscopes .....</i>	<i>3</i>
2.2	UWB SENSORS .....	3
2.2.1	<i>Ranging techniques .....</i>	<i>3</i>
2.2.2	<i>Ranging implementation in DecaRanging demo .....</i>	<i>5</i>
2.3	MULTILATERATION .....	6
2.4	SENSOR FUSION .....	6
2.4.1	<i>Xsens estimation engine .....</i>	<i>6</i>
2.5	ALLAN VARIANCE .....	7
<b>3</b>	<b>METHOD .....</b>	<b>8</b>
3.1	HARDWARE .....	8
3.1.1	<i>Xsens MTi-300 sensor .....</i>	<i>8</i>
3.1.2	<i>Decawave, DecaRanging demo card .....</i>	<i>8</i>
3.1.3	<i>Raspberry Pi single board computer .....</i>	<i>8</i>
3.1.4	<i>Sensor platform .....</i>	<i>8</i>
3.2	SOFTWARE .....	10
3.2.1	<i>UWB board listener application .....</i>	<i>10</i>
3.2.2	<i>UWB board ranging application .....</i>	<i>10</i>
3.2.3	<i>Mti-300 sensor data application .....</i>	<i>11</i>
3.3	LAB SETUP .....	11
3.3.1	<i>Preparations .....</i>	<i>11</i>
3.3.2	<i>Measurements .....</i>	<i>12</i>
<b>4</b>	<b>RESULTS .....</b>	<b>13</b>
4.1	MEASUREMENTS .....	13
4.1.1	<i>Long run .....</i>	<i>13</i>
4.1.2	<i>Rotation .....</i>	<i>15</i>
4.1.3	<i>Translation .....</i>	<i>17</i>
4.1.4	<i>Combined .....</i>	<i>20</i>
<b>5</b>	<b>CONCLUSION AND DISCUSSION .....</b>	<b>28</b>
5.1	ERROR SOURCES .....	29
5.2	SUGGESTIONS FOR FURTHER DEVELOPMENT .....	29
<b>6</b>	<b>REFERENCES .....</b>	<b>30</b>
<b>7</b>	<b>APPENDIX .....</b>	<b>31</b>
7.1	KNOWN ISSUES .....	31
7.2	UWB MESSAGE FORMAT .....	32
7.2.1	<i>IEEE 802.15.4 UWB physical layer .....</i>	<i>32</i>



7.2.2	IEEE 802.15.4 MAC layer .....	32
7.3	RASPBERRY PI SETUP .....	33
7.3.1	General setup .....	33
7.3.2	Wi-Fi setup .....	33
7.3.3	Preparing for the UWB sensor .....	34
7.3.4	Preparing for the IMU sensor.....	34



# 1 Introduction

## 1.1 Background

In order to determine the orientation and position of an object it is common to measure linear and angular motion with the help of an inertial sensor, or inertial measurement unit (IMU). Motion is measured with the help of on board accelerometers, gyroscopes and possibly also magnetometers. Orientation and position are then determined by integration of the motion data. Traditionally used in the aviation and marine industry, when their relative size and price made them suited for little else, IMUs can today be found in cars, gaming consoles, smart phones and in all sorts of devices that uses motion tracking. This have been made possible mainly thanks to micro machined electromechanical systems (MEMS), microprocessors interacting with micro sensors, making the modern day sensors both smaller and cheaper to manufacture than their predecessors. The decrease in size and price comes at the cost of reduced performance in accuracy and stability, and it can therefore be desirable to aid the IMU with additional sensors, like an ultra wideband sensor.

Ultra-Wideband (UWB) is a radio technology for transmitting information using bandwidths of 500 MHz or more. A fairly large frequency band compared to the kHz range of traditional radio communication. This makes it possible to effectively use scarce radio bandwidth while at the same time enabling high data communication. UWB communication is by industry standards limited in its output power. Because of this the usual UWB applications are of short range, indoor type, in environments where the UWB device needs to coexist with other radio devices or in radio frequency sensitive environments. Other than high data rate transfers, UWB also often also uses pulse radio with very short pulses, giving a high spatial resolution. This coupled with a bandwidth that helps avoid multi-path propagation makes UWB well suited for localization application

## 1.2 Purpose

The purpose of this study is to evaluate a sensor fusion of the *Xsens MTi-300* IMU and the *DecaWave-1000* (DW1000) UWB sensor in order to provide means to determine the position of a mobile robot platform. This includes setting up a sensor platform for data acquisition, configuring the platform to provide data in a desired way and finally fusing and filtering the sensor data to acquire position. There is currently insufficient data on the DW1000 performance, while the MTi-300 can be considered well tested. Extensive tests are therefore needed on the DW1000 sensor alone. The goal is to provide positioning with sensor fusion that is better than either of the sensors running separately.

### 1.2.1 The robot platform

The robot meant to utilize the solution is an industrial type four-wheeled platform capable of autonomous navigation. The sensor platform in this study will be developed as a standalone module and will be able to run independent of the sensors that are currently in the robot platform, with the future possibility of integrating the two systems to utilize all sensors in one sensor fusion solution, or by using only the sensor platform for positioning. The planned setup of the robot platform is outlined in the black elements in figure 1. The sensor platform module

consists of an IMU and a UWB device, henceforth referred to as a *tag node*, controlled by a Raspberry Pi computer. The sensor platform acquires IMU and UWB sensor data and transfers and transmits the data via Wi-Fi. The data is then processed separately from the robot platform. Apart from the processing PC, the only operating elements outside of the robot platform are the stationary UWB devices nodes used in the UWB ranging system, henceforth referred to as *anchor nodes*.

Positioning of the current robot platform is based solely on visual positioning, with a PC providing user specified coordinates and an on-board processor determining motor control signals based on current position and desired position. These parts are illustrated in the grayed out elements in figure 1.

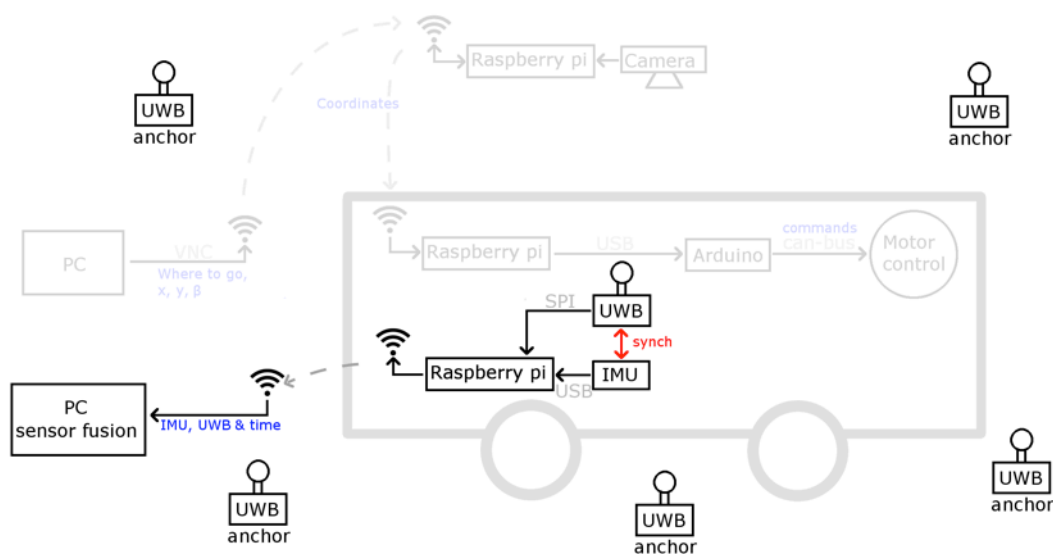


Figure 1: The robot platform with the sensor platform highlighted. A raspberry Pi computer gathers data from the two synced sensors and sends it to an external PC for data processing.



## 2 Theory

### 2.1 IMU sensors

Typically three orthogonal accelerometers are used for measuring linear acceleration and three gyroscopes measure the angular rate of change in order to determine position and orientation in 3 space. In positioning with IMU, a critical cause of drift is caused by white noise perturbing gyroscope signals [1] which can be helped significantly by performing a sensor fusion using magnetometers. Therefore it is not uncommon to incorporate a magnetometer in the IMU solution. The IMU is self-contained, in that it does not rely on external units for its measurements.

#### 2.1.1 IMU accelerometers

MEMS inertial accelerometers consist of a mass-spring system where the spring and mass are situated in a vacuum. Mass displacement is measured by a sensor that gives a signal proportional to the mass displacement. Thus the forces that are acting on the system and are parallel with the spring axis can be measured. With knowledge about the system mass the system acceleration in the direction of the spring axis can be determined using Newton's second law.

#### 2.1.2 IMU gyroscopes

MEMS gyroscopes work with a mass that oscillates in the kHz range. As in the case with the accelerometer, mass displacement is measured with an output signal that is proportional to the mass displacement perpendicular to its axis of oscillation. When the gyroscope is rotated the mass experiences a Coriolis force that is larger when the mass is further away from the center of rotation, and by comparing the signal readout on either side of the oscillating movement rate of turn can be determined.

### 2.2 UWB sensors

#### 2.2.1 Ranging techniques

One ranging technique used to determine the distance between two node points is time of flight (TOF) measurement. That is, measuring the time it takes for an electromagnetic wave to travel from one node to another. If a TOF  $\tau_{tof}$  is known then the distance  $d$  between two nodes is determined by

$$d = \tau_{tof} \cdot c \quad (1)$$

where  $c$  is the speed of light.

There are several methods to determine the TOF between two nodes. Two of these, one-way Time-of-arrival and two-way Time-of-arrival ranging, are described below.

### 2.2.1.1 One-way Time of Arrival (TOA) Ranging

Assuming a system with a pair of nodes 1 and 2 whose clocks are perfectly synchronized. Node 1 sends a message containing the time  $\tau_1$  when the message was sent. When node 2 receives the message at time  $\tau_2$ , the TOF is determined by

$$\tau_{tof} = \tau_2 - \tau_1 + \delta + e \quad (2)$$

Where  $\delta$  is a possible delay caused by multipath propagation or non line of sight conditions and  $e$  is i.i.d. Gaussian noise.

This method is simple and only requires one single message to be sent, but is completely dependent on synchronization.

### 2.2.1.2 Two-way Time of Arrival Ranging

In a system with a pair of nodes 1 and 2 whose clocks are not synchronized, two-way ranging can be used to allow the system to estimate the round trip time. At a time  $\tau_1$  node 1 transmits a message which node 2 receives at time  $\tau_2$ . Node 2 sends a response message at time  $\tau_3 = \tau_2$  which is received at time  $\tau_4$  by node 1. Finally node 1 sends the final message used in the ranging at time  $\tau_5$  which is received at time  $\tau_6$ . Round trip times  $\tau_{RT1}$  and  $\tau_{RT2}$  are calculated by removing the response delays  $\tau_{d1}$  and  $\tau_{d2}$  from each of the round trips,

$$\tau_{d1} = (\tau_3 - \tau_2) \quad (3)$$

$$\tau_{d2} = (\tau_5 - \tau_4) \quad (4)$$

$$\tau_{RT1} = (\tau_4 - \tau_1) - \tau_{d1} \quad (5)$$

$$\tau_{RT2} = (\tau_6 - \tau_3) - \tau_{d2} \quad (6)$$

Thus the TOF is

$$\tau_{TOF} = \frac{(\tau_{RT1} + \tau_{RT2})}{4} + \delta + e \quad (7)$$

Where  $\delta$  is a possible delay caused by multipath propagation or non line of sight conditions and  $e$  is i.i.d. Gaussian noise.

In comparison to one-way TOA ranging, synchronization between nodes is not important and can therefore be preferable when synchronization is impossible or hard to achieve with good precision.

Even though synchronization is not needed, errors can still occur if the clock drift is different on the two nodes. In a typical indoor ranging, the TOF is in nanosecond order while the delays  $\tau_{d1}$  and  $\tau_{d2}$  can be in order of micro or milliseconds. Therefore the error in  $\tau_{tof}$  can grow big if the relative drift differs between the nodes.

### 2.2.2 Ranging implementation in DecaRanging demo

In the DecaRanging demo, two UWB devices can range with each other by configuring them as tag node and anchor node respectively, using physical switches on the PCBs. The two way ranging algorithm is illustrated in figure 2.

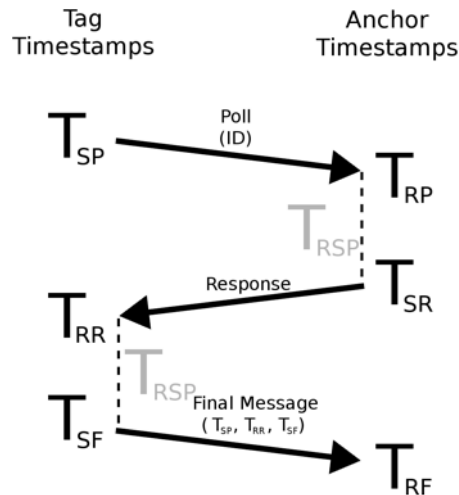


Figure 2: A two way ranging algorithm implemented in DecaRanging demo.

Once powered up, a tag send out broadcasts, or blinks, with one-second intervals. When the anchor receives a blink, it sends back a ranging initiation message that is received by the tag, which then initiates the two-way ranging. The tag sends a poll message to the anchor that responds with a response message. The ranging is completed when the tag then sends a final message containing all the relevant timestamps, followed by a sleep time of 400 ms before starting over by sending a new poll. After receiving the final message, the anchor calculates the TOF and between the two nodes. This distance is affected by a range bias which needs to be compensated for with a certain offset. The offset that is used follows a table of values, illustrated in figure 3.

After calculating the distance it is displayed on the onboard LCD screen. In demo mode, a final report message is also sent from the anchor node for displaying the same information on the tag node LCD screen.

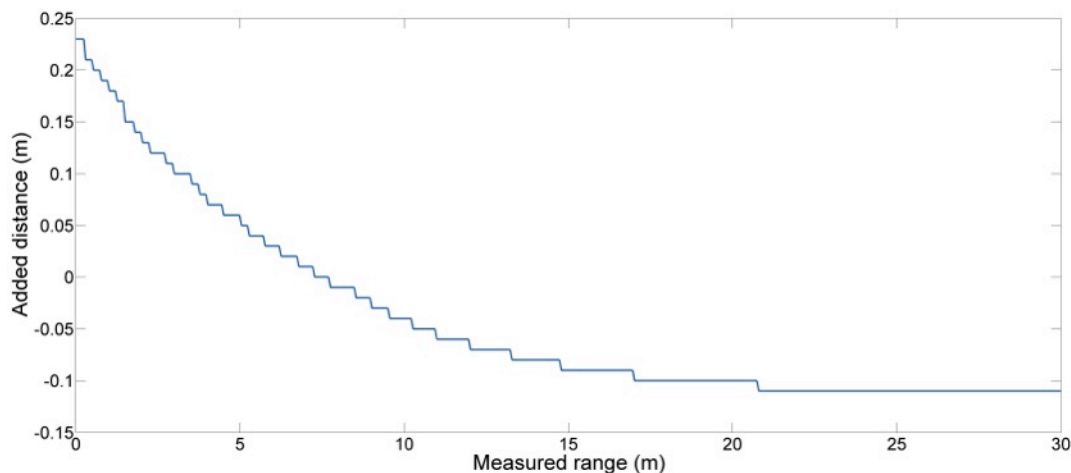




Figure 3: Range bias adjustment. Offset added as a function of measured distance.

## 2.3 Multilateration

Multilateration is a technique that uses difference in distance between anchor nodes with known position and a tag node to determine the position of the tag. Distance between only one pair of nodes gives an infinite amount of possible positions along a curve; hence to get an absolute position multiple measurements are needed.

Given that the positions  $\mathbf{A}_i = (x_i, y_i, z_i)$  of  $m$  anchor nodes and the distances  $d_i$  from anchor node  $i$  to the tag node are known, where  $i = 1, 2, \dots, m$ . Then, if the coordinates of the tag node is  $\mathbf{x} = (x, y, z)$  then the following equation hold

$$\|\mathbf{A}_i - \mathbf{x}\|_2 = d_i \quad (8)$$

The only unknown parameter in equation (8) is the coordinate vector  $\mathbf{x}$ . Assuming that the noise affecting the system is Gaussian, the estimated coordinate vector  $\hat{\mathbf{x}}$  can be determined by solving a *least squares problem*, given by

$$\hat{\mathbf{x}} = \min_{\mathbf{x}} \sum_{i=1}^m \frac{1}{2} \|r_i(\mathbf{x})\|_2^2 \quad (9)$$

where  $r_i(\mathbf{x})$  is the residual function given by

$$r_i(\mathbf{x}) = \|\mathbf{A}_i - \mathbf{x}\|_2 - d_i, \quad i = 1, 2, \dots, m \quad (10)$$

This problem can be solved using standard iterative optimization algorithms [2]. These use a starting point  $\mathbf{x}_0 = (x_0, y_0, z_0)$ , which is improved in each iteration until a local minimum is found.

## 2.4 Sensor fusion

Sensor fusion deals with the problem of combining measurements from different sensors to get a result that is better than what each of the individual sensors can produce. This problem can be divided into different state estimation problems. In the case where the available information is the previous measurement data up to the present time and the current state is to be estimated, it is known as a filtering problem.

### 2.4.1 Xsens estimation engine

Xsens estimation engine (XEE) is an optimization framework for formulating and solving a wide range of estimation problems. Given measurements  $\mathbf{z}_i$  and unknown variables  $\mathbf{x}$ , a general optimization problem with equality constraints is given as

$$\arg \min_{\mathbf{x}} \sum_{i=1}^N f_i(\mathbf{x}, \mathbf{z}_i) \quad (11)$$

$$\text{subject to } c_i(\mathbf{x}, \mathbf{z}_i) = 0, \quad i = 1, \dots, M \quad (12)$$

where  $f_i$  are a number of additive cost functions to be minimized and  $c_i$  are a number of equality constraints. By handling the estimation as an optimization problem, the user can handle a sensor fusion by adding cost functions for each sensor.

XEE has support for the necessary cost functions needed to get position and orientation states using IMU sensor measurements. Hence, in the case of fusion using IMU and UWB measurements, only the cost function in equation (9) needs to be implemented. Adding the available measurements in every time step gives a tightly coupled sensor fusion, as illustrated in figure 4.

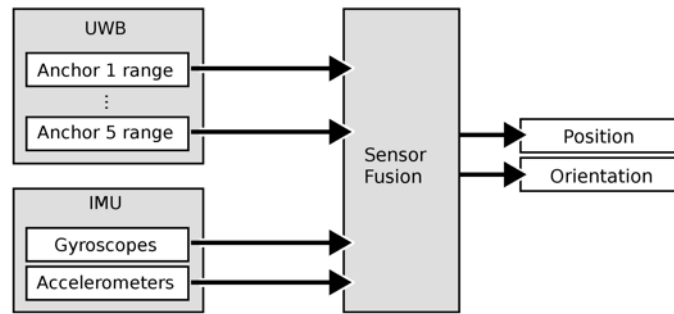


Figure 4: Tightly coupled sensor fusion using IMU and UWB data.

## 2.5 Allan variance

Allan variance is a method to determine underlying noise characteristics in signals. By taking an array of data and dividing it into  $n$  buckets of length  $t$  and taking the average value  $a$  of every bucket, Allan variance is computed as

$$AVAR(t) = \frac{1}{2(t-1)} \sum_{i=1}^{t-1} (a(t_{i+1}) - a(t_i))^2 \quad (12)$$

Allan deviation, defined as the square root of the Allan variance can then be plotted as a function of average time  $t$  on a log-log scale. Different types of random processes can then be identified and their numerical parameters read directly from the plot. Processes of interest for UWB ranging are random walk and bias instability.

At short averaging times, Allan variance is dominated by noise in the sensor. Random walk can be identified by finding the initial slope in the plot that has a gradient of -0.5, and fitting a line to the slope and taking its value at  $t = 1$ .

As average time increases, bias instability can be identified as the region where the Allan deviation has its minimum. The value in this point is the limit of precision, signifying the inherent instability in the sensor output.

## 3 Method

### 3.1 Hardware

#### 3.1.1 Xsens MTi-300 sensor

The MTi-300 contains an IMU sensor assembly, as described in the theory section, combined with a 3D magnetometer and a barometric pressure sensor. Using on board processors it is able to provide processed sensor data making it a full gyro-enhanced attitude and heading reference system (AHRS). For the sake of this study, only unprocessed data from the accelerometers and the gyroscopes will be used.

Communication is handled via a 9-pin Molex connector of PicoBlad type. For synchronization purposes, pin 5 needs to be accessed separately, but the other pins of interest can be used as a regular USB bus device. A CA-USB2G cable adapter provides syncIn access as well as the Molex↔USB conversion.

#### 3.1.2 Decawave, DecaRanging demo card

EVB1000 from DecaWave is an evaluation circuit board incorporating the DW1000 UWB wireless transceiver IC, an ARM type Cortex M3 processor and a wideband planar omnidirectional antenna. It allows communication via USB and via SPI bus interfaces. EVB1000 also comes pre-installed with ‘DecaRanging’, a two-way ranging demo application. The manufacturer recommends the two planes of the antennae of two ranging devices to be parallel to each other for best results.

Since USB communication with the DW1000 requires the on board ARM processor to convert the SPI bus communication to USB, making communications inherently slower, the preferred means of communication are the SPI pins.

#### 3.1.3 Raspberry Pi single board computer

The Raspberry Pi is a single board computer measuring 18x56mm. It has a system on a chip that includes a 700Mhz ARM type 11 processor, a VideoCore IV GPU and 512MB of SDRAM. Among its communication ports it has 8 GPIO pins, an SPI bus with two chip selects and 2 USB 2.0 ports. It is running the operating system Raspian, an ARM version of the operating system Debian, built on the Linux kernel, written especially for the Raspberry Pi.

Outlined in the appendix (7.3) are the modifications that have been made to the software that comes provided with the Raspberry Pi used in this study.

#### 3.1.4 Sensor platform

The sensor platform runs all sensors and performs data acquisition and storing. For purpose of mobility the sensor platform needs to be fairly small, compact and self powered. For these reasons, and the ability to communicate with all sensors, the Raspberry Pi computer is used as a sensor platform controller. A general layout is illustrated in figure 5.

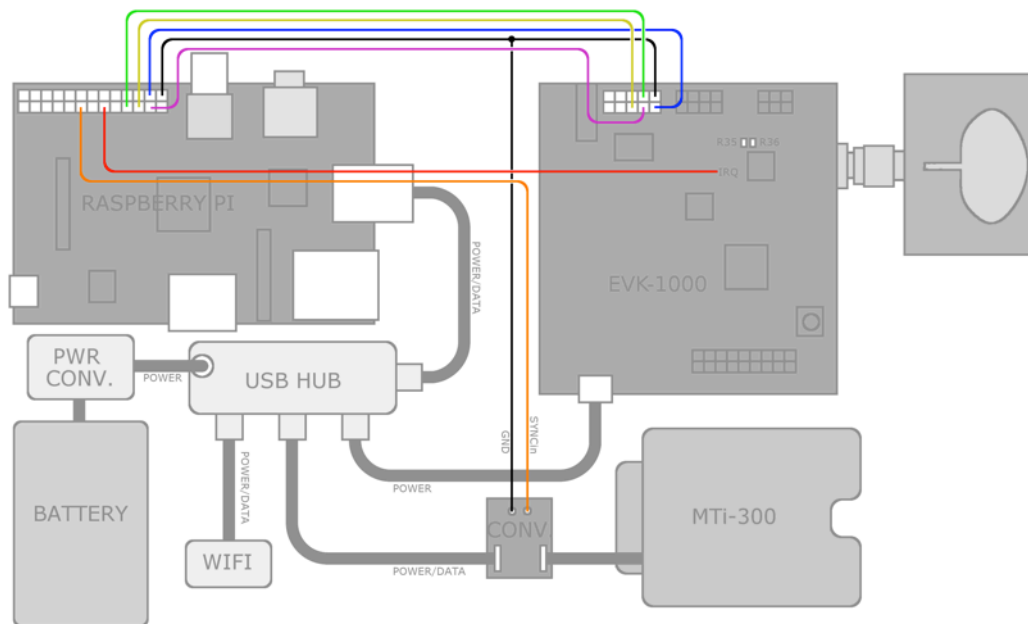


Figure 5: Sensor platform layout. Colored lines illustrate wiring for SPI communication, UWB↔IMU sync and interrupt handling.

The platform is powered through a regular powered USB hub. A 15V Battery is connected via a power converter and provides 5V DC to the hub. The hub in turn powers the MTi sensor, the UWB sensor, a wireless network receiver and the Raspberry Pi. Since Raspberry Pi from revision 2 allows power feedback on its USB data ports, and the Raspberry Pi used in the platform is a revision 7, the USB hub is used to power the Raspberry Pi through the standard USB data port rather than by conventional means. The Raspberry Pi is otherwise meant to be powered solely via the designated power input micro USB port. The USB hub also serves the purpose of transferring data between the Raspberry Pi and the MTi sensor as well as between the Raspberry Pi and the wireless network receiver.

4 GPIO pins providing SPI communication on the UWB board are connected to their equivalent GPIO pins on the Raspberry Pi. The connections are further described in table 1.

Table 1: Pin connections between the different PCBs.

	Raspberry Pi, GPIO	connections
<b>MOSI</b>	pin 10	UWB J6, pin 08
<b>MISO</b>	pin 09	UWB J6, pin 05
<b>SCLK</b>	pin 11	UWB J6, pin 07
<b>CE0/SS1</b>	pin 08	UWB J6, pin 09
<b>IRQ</b>	Pin 19 (IN)	UWB, IRQ
<b>Sync</b>	Pin 17 (OUT)	IMU, pin 5

To avoid polling when waiting for the UWB board to receive a message, the Raspberry Pi utilizes interrupts. The interrupt request (IRQ) signal pin of the DW1000 is connected to



GPIO pin 19 on the Raspberry Pi, configured as an input pin. For synchronization with the IMU, the Raspberry Pi GPIO pin 17 is configured as an output pin and is connected to SyncIn pin 5 of the IMU, exposed with a USB<->MOLEX adapter between the Mti-300 and the USB hub. This is used to signal the IMU SyncIn pin with a 3.3V pulse. Finally, all three PCBs are connected to a mutual ground reference point.

### **3.2 Software**

Two programs have been written in C++ for the DW1000 with a raspberry Pi as their host system; a listener application with the purpose of picking up and parsing UWB messages and a ranging application for positioning purposes. Both programs use interrupt handlers to respond to IRQ sent from the DW1000. The DW1000's IRQ is configured to be set when receiving a message. For IMU data, a program has been written to log raw data output.

#### **3.2.1 UWB board listener application**

The listener application picks up the transmissions of other UWB devices and parses the received data. Information about the structure of the messages can be found in Appendix (7.2). The interrupt handler calls a function that reads the received message information and payload. The destination and source addresses are then displayed along with an eventual payload and message type.

#### **3.2.2 UWB board ranging application**

The ranging application provides the Raspberry Pi with the ranging data between one tag node and any desired number of independent anchor nodes. The anchor nodes are running the standard DecaWave DecaRanging demo that is described in the theory section. By using the preinstalled software on the anchors, only the tag needs to be connected to a host system. Similar to the listener, the tag node sleeps until woken up by an interrupt. It utilizes a state machine that is dependent on the type of message received. With the anchors running the standard DecaRanging demo routines the state machine closely follows the two-way ranging communication method described in the theory section.

Given that the number of anchor nodes available for ranging is at least the amount specified by the user, ranging initialization messages containing information on which delay time to use are sent to all the anchor nodes, before proceeding to the ranging algorithm.

After the ranging has been initialized, the program follows the same ranging steps as the DecaRanging. As soon as a poll message is received and a response message in turn has been sent, the tag disregards other messages until either a final message from the anchor is received, or a specified amount of time has passed in which case a time out occurs and polls are allowed again. In the case where the final message is properly received, the distance is calculated using all the necessary timestamps. Saved to a log file are then the distance, the timestamps and the corresponding anchor id. Whenever a ranging is completed, GPIO pin 17 on the raspberry Pi is set as a step in synchronizing the UWB and the IMU sensor measurements.

One drawback of using the DecaRanging on the anchor nodes is that there is a sleep time of 400 milliseconds programmed after each ranging. This reduces the maximum speed at which range measurements can be obtained to around 2 Hz per anchor.



### 3.2.3 Mti-300 sensor data application

The MTi-300 sensor data application is a C++ script that consists of two functions for running the IMU sensor in the sensor platform. A startup function scans computer USB ports to find the MTi-300 (an *MTi mark IV* device). After identification it configures the device to record raw data with time stamps to a log file on the host computer with 400Hz sampling rate. It also configures the device to register received sync pulses in the log file. A stop function ends the logging and shuts down and disconnects the device from the computer.

## 3.3 Lab setup

### 3.3.1 Preparations

The tests that are performed with the sensor platform aims mainly to establish the behavior of the UWB sensor and the synchronization between the UWB sensor and the IMU. A final test will determine the overall effectiveness of the final solution. Two different anchor node setups are used, as illustrated in figure 4 and figure 5.

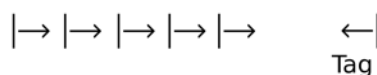


Figure 6: Positioning of the anchor nodes relative the tag node in node setup 1.

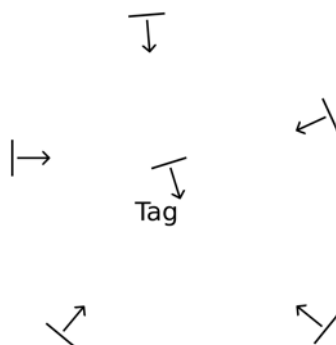


Figure 7: Positioning of the anchor nodes relative the tag node in node setup 2

Node setup 1 has all UWB sensors in a straight line and is used to test specific UWB antenna effects. The anchor nodes are evenly spaced within roughly 1.5m and their antennas are facing the sensor platform. The sensor platform antenna is facing the anchor nodes and has a clear line of sight to all anchor nodes.

Node setup 2 has the anchor nodes positioned in a ring around the tag node to resemble a realistic robot platform-positioning scenario. In this setup the anchor nodes are evenly spaced within roughly 3x3m in reference system axis x and y and 1.5m in reference system axis z and their antennas are facing the center of the ring. The sensor platform antenna has a clear line of sight to all anchor nodes.

For reference, all tests except the long run measurement will be recorded with a high-resolution visual tracking system consisting of five high resolution IR cameras triangulating the position of light reflecting markers. This reference system has one marker on the sensor platform antenna, 3 markers on orthogonal axes placed on the side of the reference system, making it possible to measure sensor platform rotation, and one marker on each of the five



anchors. Since the system provides high precision anchor positioning it also serves as the global reference system that all measurements uses. When using the reference system it is the first system to be turned on and the last system to be turned off in order to ensure full reference recording of the measurement. Since the UWB ranging software controls the sync pulses that are sent to the IMU sensor the UWB ranging software needs to be initiated before the IMU sensor software. For the same reasons, when ending a measurement the IMU needs to be turned off before the UWB ranging software.

After initiating a measurement, with all sensors running, the sensor platform is excited briefly to register a short pulse in the IMU angular velocity data. This pulse is later compared with calculated angular velocity from the reference system positioning data. By measuring the data stream correlation between IMU and reference system a data stream offset term can be acquired and used to sync the reference system data stream with the sensor platform data stream.

### **3.3.2 Measurements**

The measurements are divided into five different types. The first four types are meant to evaluate the performance of the UWB sensor. The fifth and final type is to evaluate the final filter solution for positioning using the fused sensor data from both the UWB sensor and the MTi-300.

#### *Long run*

A measurement of only the UWB sensor ranging, lasting approximately 12 hours. This will be used to evaluate the Allan variance of the UWB sensor.

#### *Static*

Similar to the long measurement, but with IMU and reference system measuring simultaneously. This will be used to measure the effect of a possible offset in the UWB sensor ranging.

#### *Rotation*

A well controlled rotation of the sensor platform with the UWB tag sensor antenna in a static position in the center of rotation. This will be used to measure effects caused by antenna direction.

#### *Translation*

The sensor platform is moved along a linear trajectory in the reference system x,y – plane in two steps. This will be used to measure the effect of a possible offset in the UWB sensor ranging and how this would behave at different ranges.

#### *Combined*

A measurement combining translation and rotation. The sensor platform performs a slow planar motion locked to the sensor system x- and y-axis. Rotation is limited to the sensor system z-axis. This will be used as a final measurement of how the sensor platform performs in situations closely resembling those that the robot platform will perform in.

## 4 Results

During the development of the sensor platform and the measurement runs, various issues were encountered that was either too extensive to overcome within the duration of the study or inherent to the hardware of the study. A full list of known issues can be found in the appendix (7.1)

### 4.1 Measurements

Every anchor has a unique 8 octet number for identification purposes. For all five anchors used in the experiments the 8<sup>th</sup> octets are unique for every anchor and can therefore be used as a shortened anchor ID. The full 8 octet anchor IDs and their 8<sup>th</sup> octet 10-base equivalents are displayed in table 2.

Table 2: Translation of the last octet of the anchor ID to a 10-base digit.

Hexadecimal full ID	10-base ID
10 04 16 20 5E 37 F9 9A	154
10 01 91 20 5D FB 0F 73	115
10 04 57 20 5E 37 FA 5C	92
10 07 40 20 5E 37 FA E0	224
10 06 44 20 5E 37 FA 56	86

The reference system described in the lab setup section provides x-, y- and z coordinates in a user specified coordinate system. Anchor position and tag position, in the coordinate system, can then be used to give a reference for range measurements provided by the UWB system.

#### 4.1.1 Long run

Figure 8 shows the raw data output from UWB ranging with 5 different anchors. Anchor 115 alone shows a slight increase in range during the first minutes of measurement. This is believed to be a result of the EVK1000 ARM processor reaching a working temperature within that time. Anchor 86 displays a couple of faulty ranges. The reason for this is unclear. Calculated standard deviation is showed in table 3. Allan variance is calculated in figure 10.

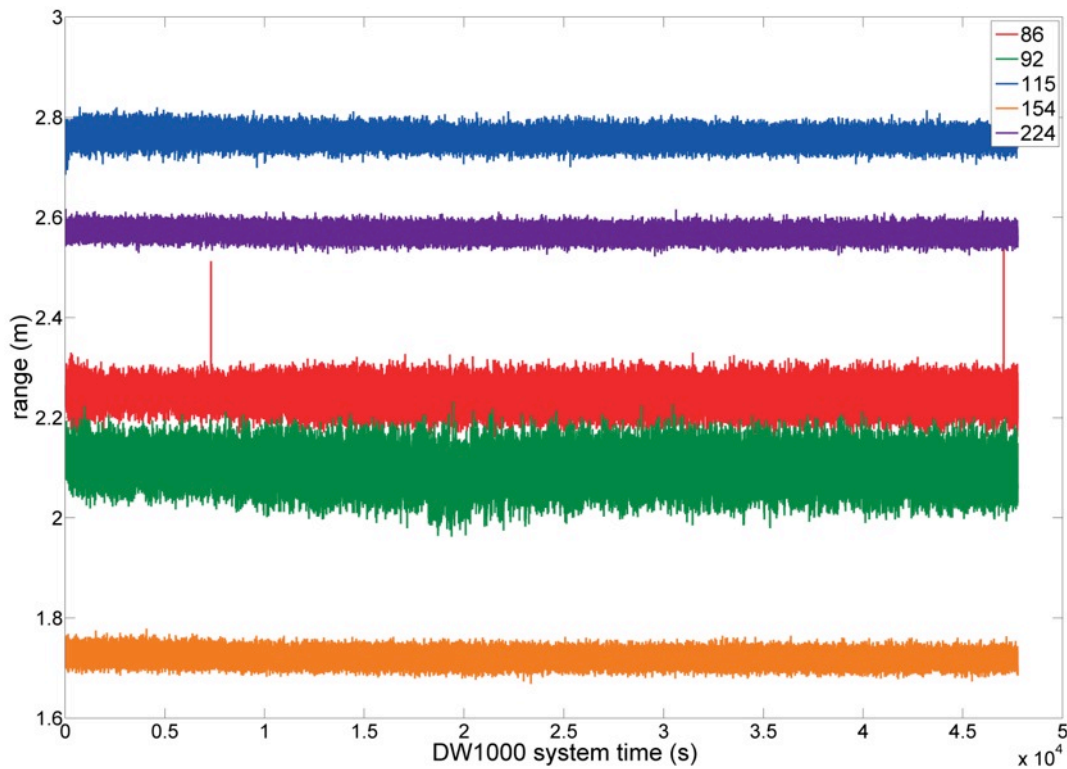


Figure 8: Ranges for the different anchors vs. measured time, during the long run measurement.

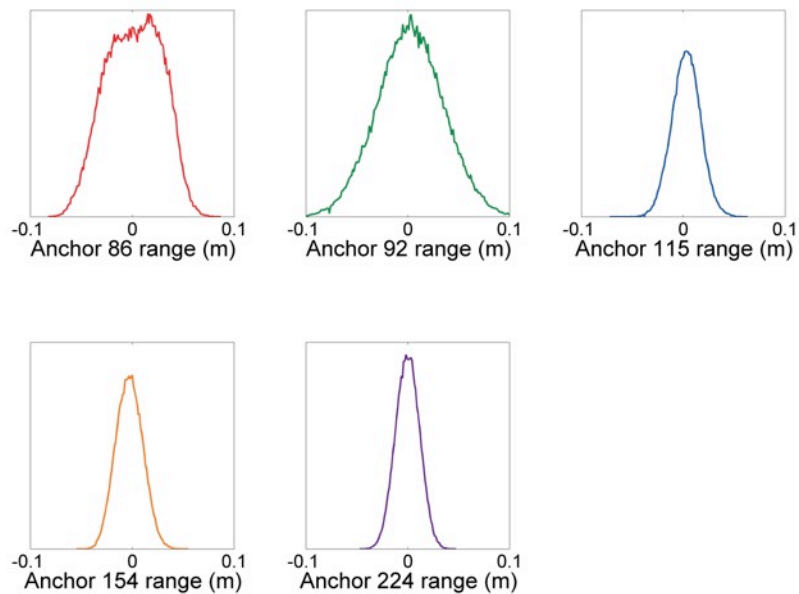


Figure 9: Histograms of the different anchors from the long run measurement. All ranges have been subtracted with their respective histogram's mean value.

Table 3: Calculated standard deviations from the long run measurements.

	Tag8 6	Tag9 2	Tag11 5	Tag15 4	Tag22 4
Standard dev. (m)	0.027	0.033	0.014	0.013	0.012

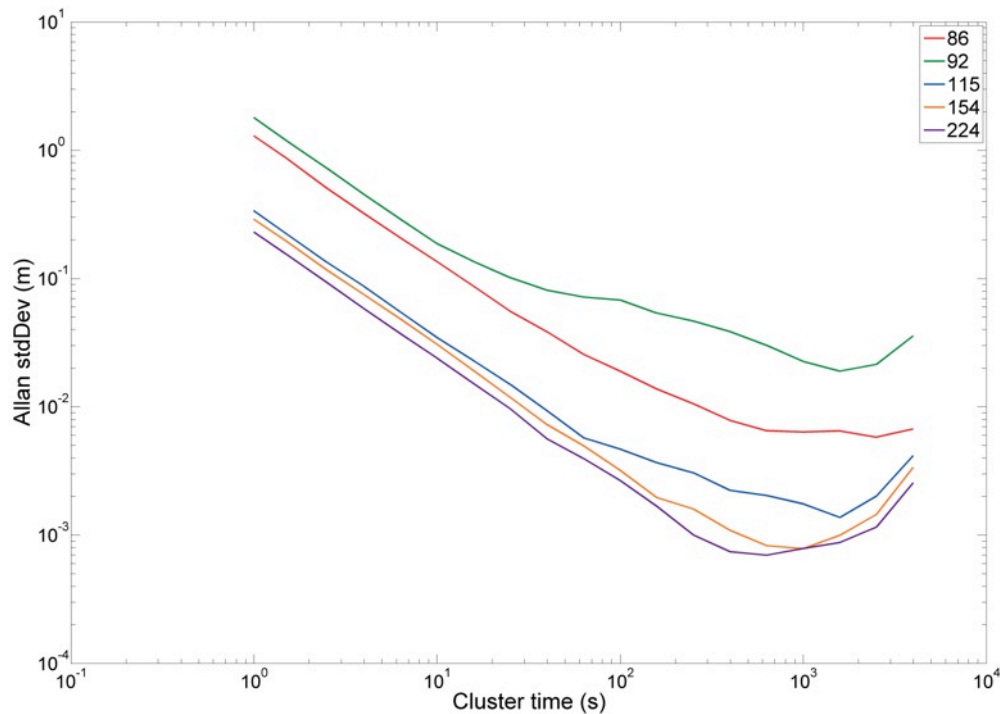


Figure 10: Allan variance of the different anchors from the long run measurement.

#### 4.1.2 Rotation

Measured ranges for the different anchors are shown in figure 11. For comparison, the mean value of every tag's measured range is plotted. The relative error between range value and mean deviation value is displayed in figure 12. Raw data from one of the gyroscopes is shown in figure 13 to illustrate the general rotational speed during the measurement. Also visible in the gyroscope data is the synchronisation excitation at the beginning and at the end of the measurement.

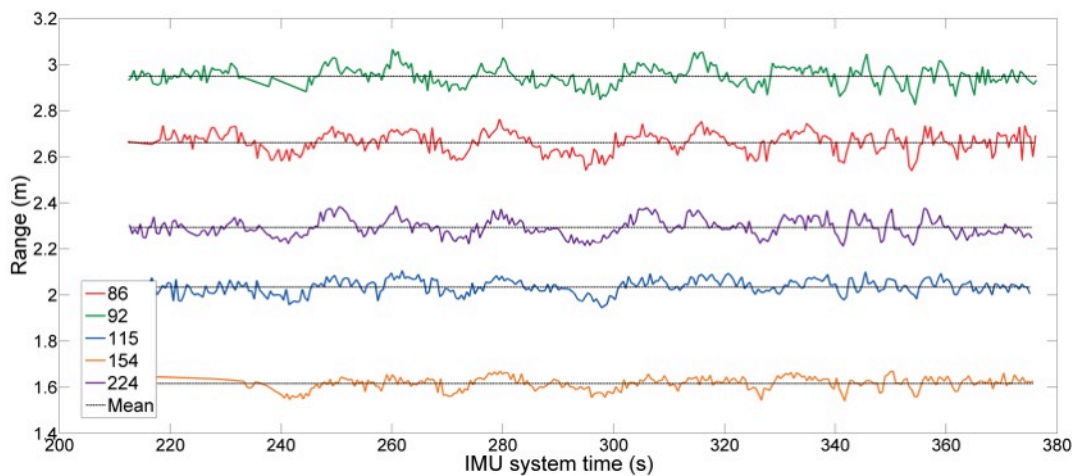


Figure 11: Range measurements of the different anchors vs. measured time. All ranges are compared with their respective mean value.

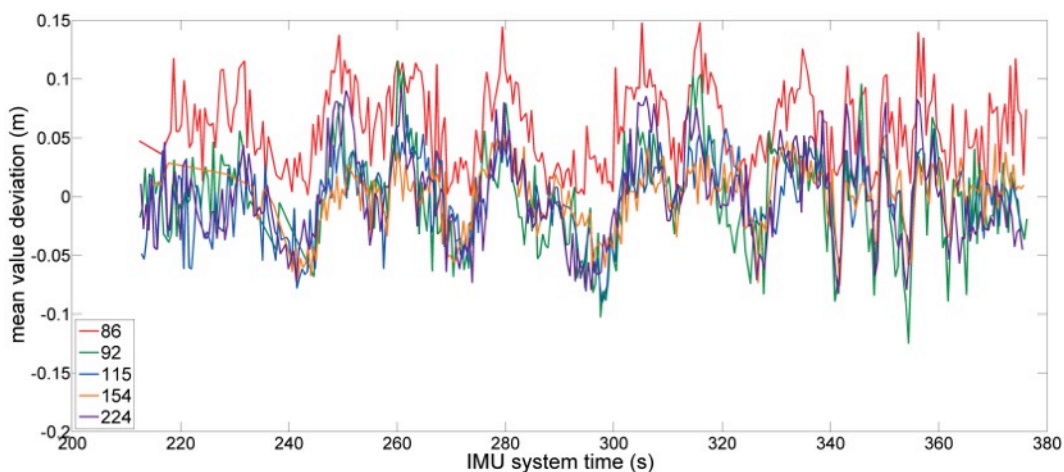


Figure 12: Deviation from mean value vs. measured time, calculated as the difference between anchor mean range value and current range value.

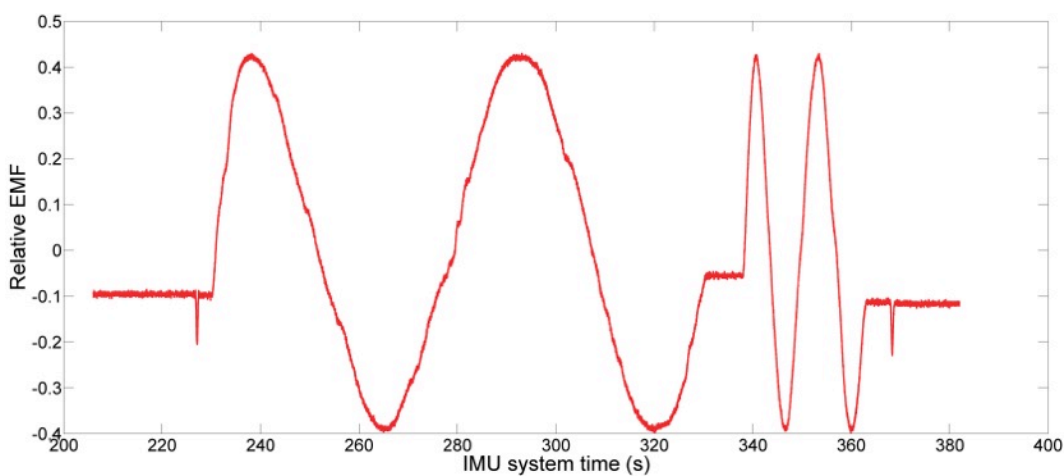


Figure 13: Raw data from one of the IMU magnetometers vs. measured

time.

#### 4.1.3 Translation

Two different measurements of translation were made. One with the anchor moving in discrete steps (figure 14, 15 and 16 respectively), resting momentarily between translations, and one with the anchor moving in one continuous motion (figure 17, 18 and 19 respectively).

Figure 14 shows the discrete translation ranges compared with the reference system and figures 15 and 16 show the range error compared to the reference system. Only anchor 115 shows any tendencies to shift error when measured distance changes. Anchor 86 and 92 shows a small positive offset from reference and anchors 154 and 224 shows a small negative offset from reference.

Figure 17 shows the continuous translation ranges compared with the reference system and figures 18 and 19 show the range error compared to the reference system. Anchors 86 and 154 seem to have an error that increases with distance. As with the discrete translation, anchors 86 and 92 shows a positive offset from reference and anchors 154 and 224 show a negative offset from reference.

Also visible in some of the range data in figure 15 and 17 is the synchronisation excitation at the beginning and at the end of the measurement.

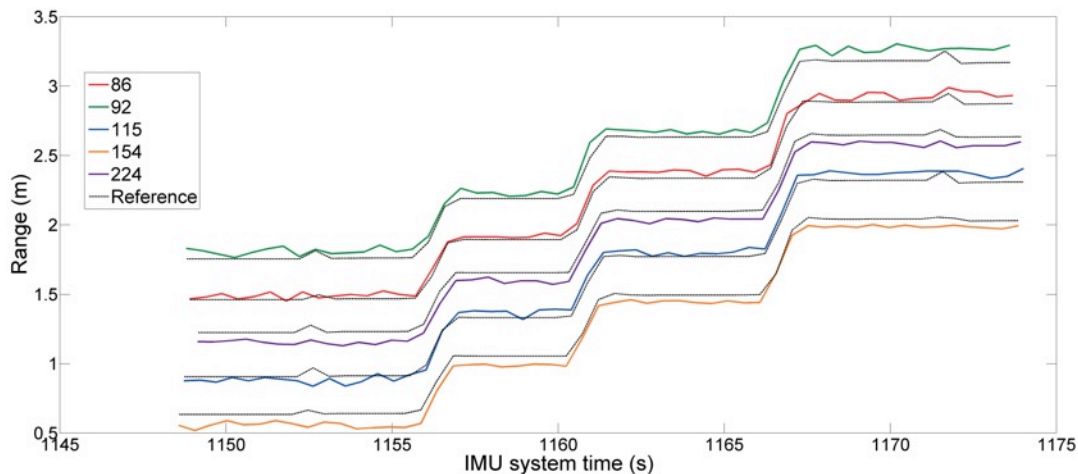


Figure 14: Range measurements of the different anchors vs. measured time. All UWB ranges are compared with their respective range from the reference system.



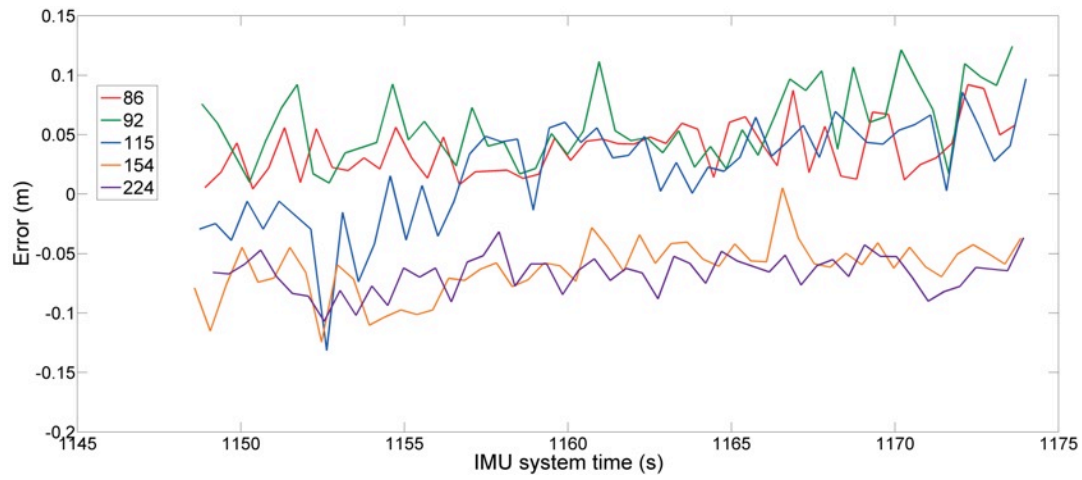


Figure 15: UWB range measurement errors calculated as the difference between UWB range value and reference system range value.

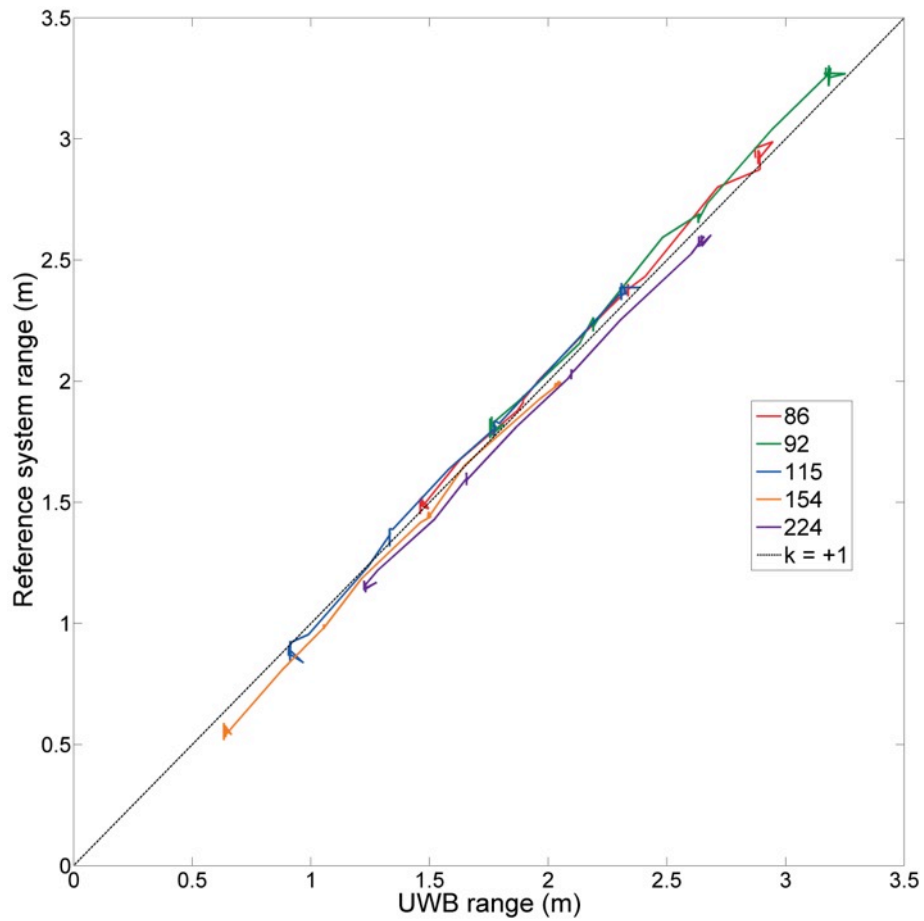


Figure 16: UWB range value vs. reference system range value.



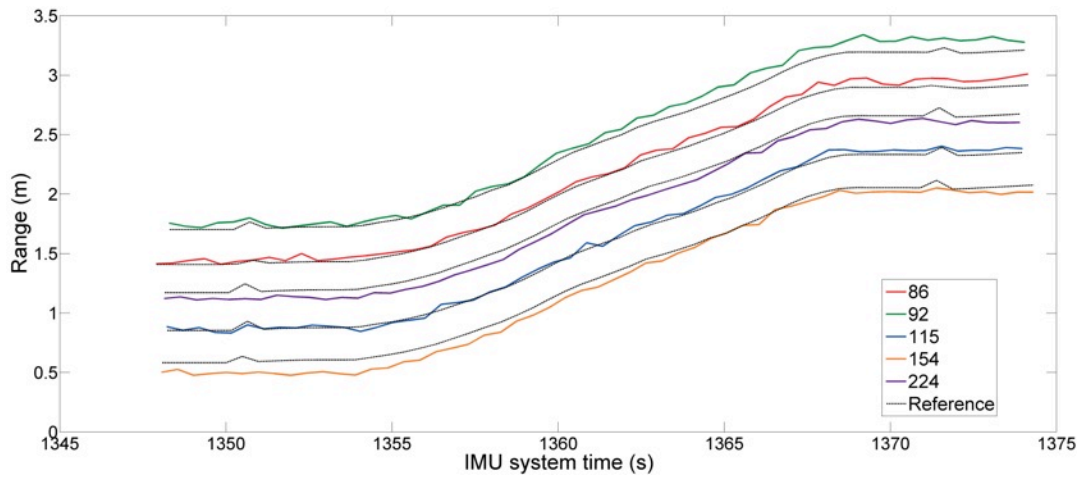


Figure 17: Range measurements of the different anchors vs. measured time. All UWB ranges are compared with their respective range from the reference system.

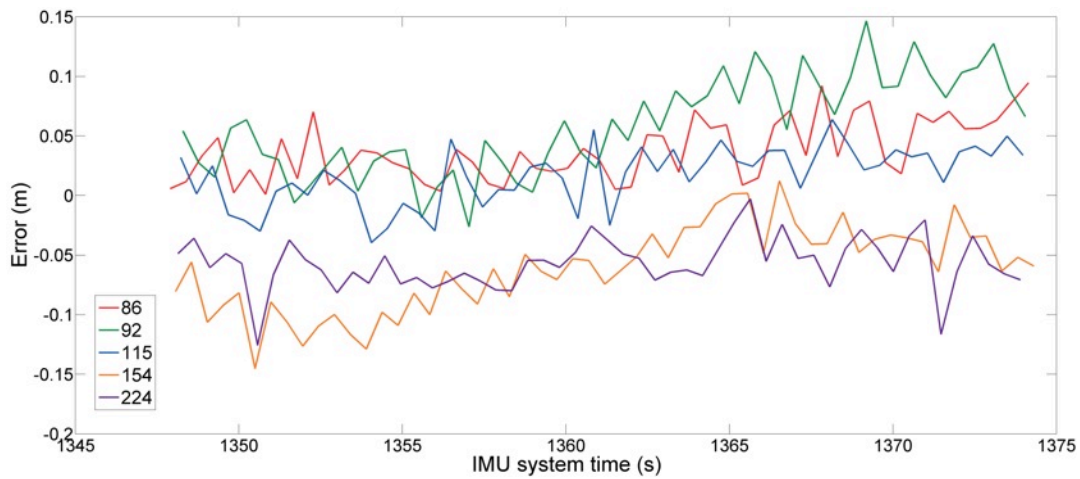


Figure 18: UWB range measurement errors calculated as the difference between UWB range value and reference system range value.

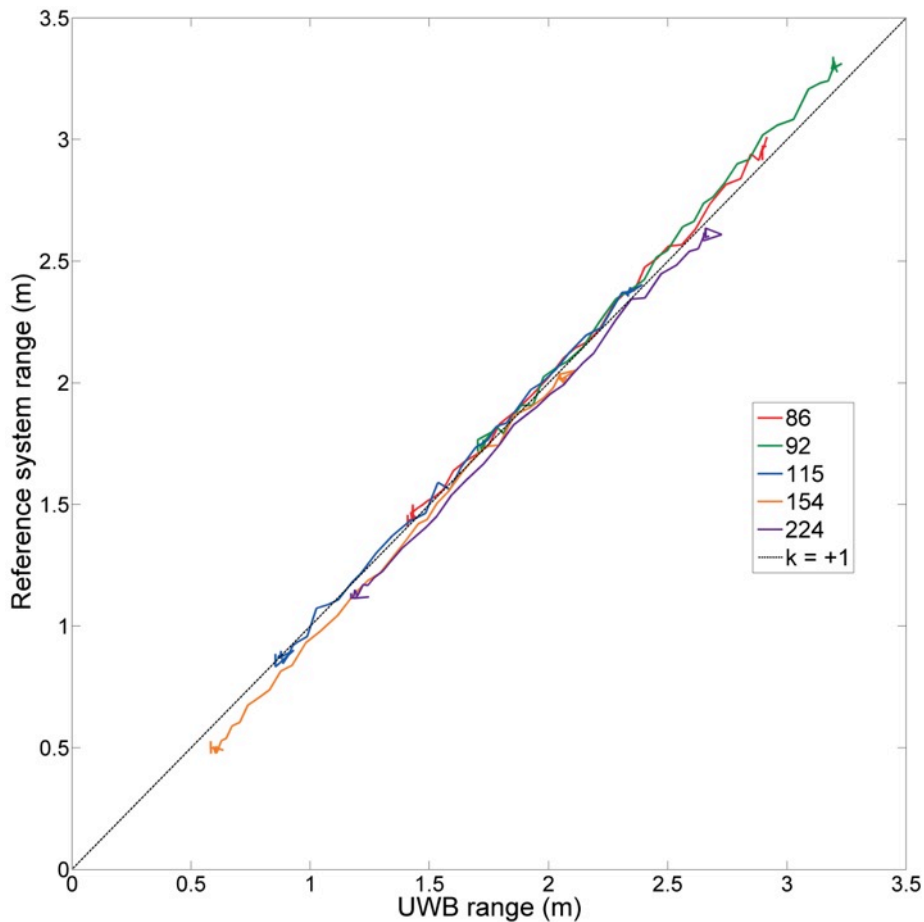


Figure 19: UWB range value vs. reference system range value.

#### 4.1.4 Combined

Results from the combined measurement can be further divided into the data gathered by the sensors (figures 20, 21, 22 and 23), the multilateration results from using only UWB data (Figures 24 and 25) and the final sensor fusion solution (Figures 26, 27, 28, 29, 30 and 31).

Figure 20 shows the raw accelerometer data. The noise-like appearance is typical for slow accelerations of the type that the robot platform will do. Also in the accelerometer data is the synchronisation excitation at the beginning and at the end of the measurement.

Figure 21 shows the gyroscope data translated to rotational values. As expected from the limits of movement of the sensor platform, the only significant changes are in the yaw angle. Also in the gyroscope data is the synchronisation excitation at the beginning and at the end of the measurement.

Figure 22 shows the continuous translation ranges compared with the reference system and figure 23 shows the range error compared to the reference system. All anchors show a positive offset at some point during the measurement. Anchor 154 at one point shows a larger than expected negative error and anchor 115 at one point shows a large positive error.

#### 4.1.4.1 Data

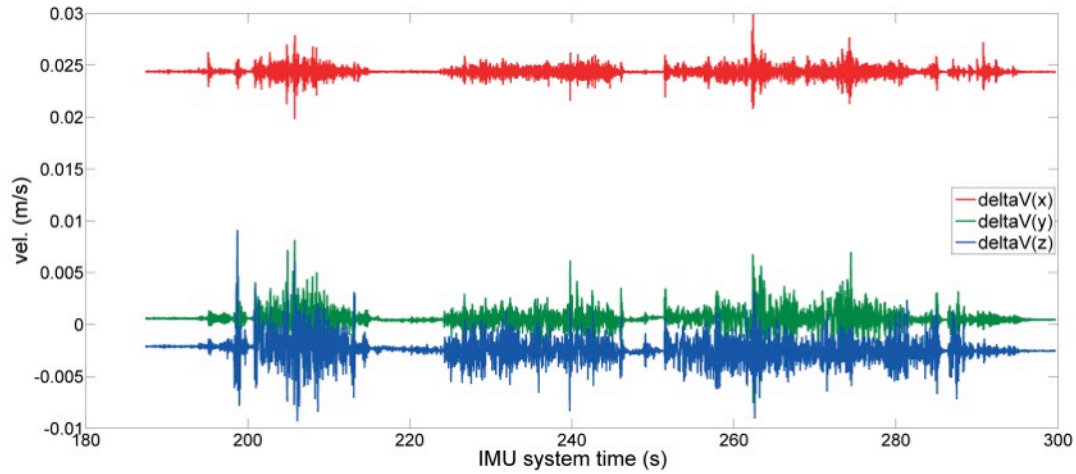


Figure 20: IMU accelerometer data vs. measured time.

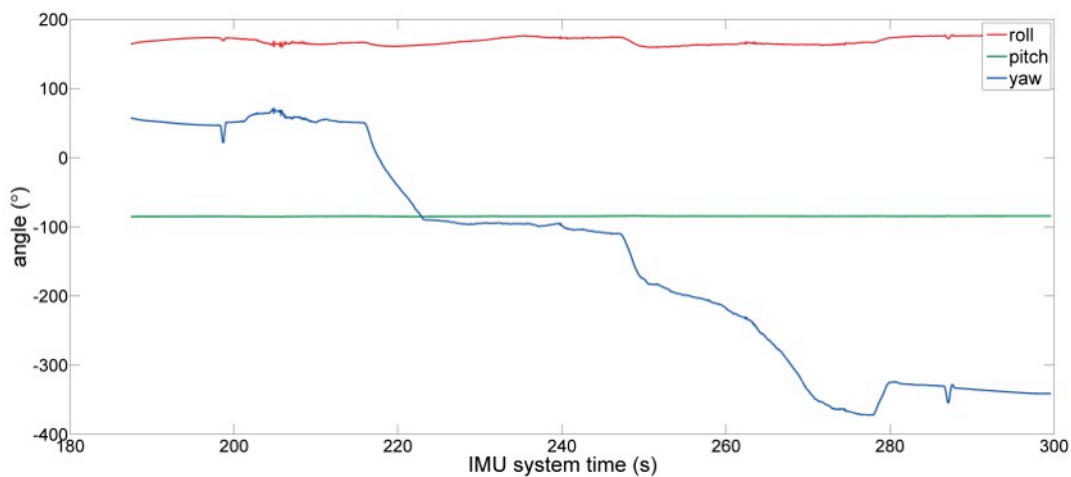


Figure 21: IMU rotation data vs. measured time

Attempts at solving for position using only IMU data yields large errors within a few seconds.

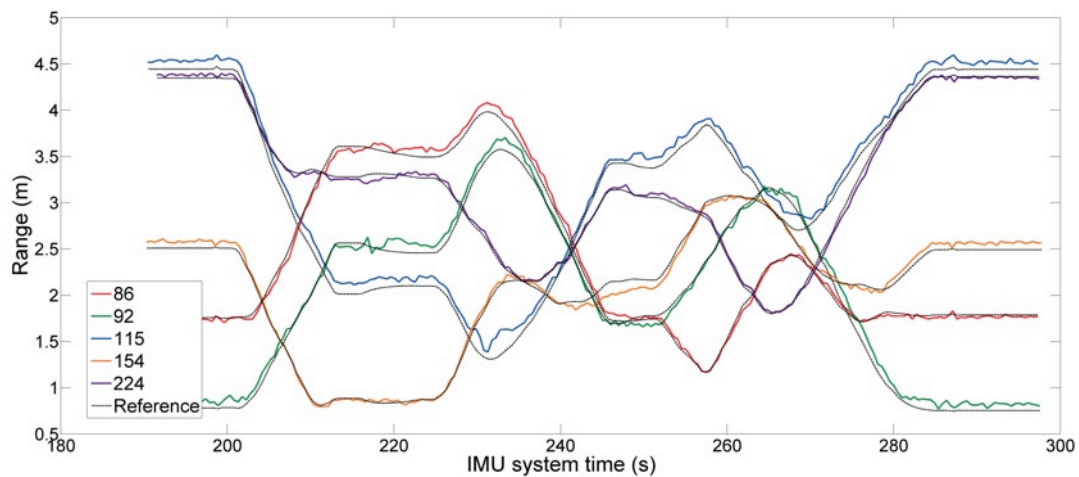


Figure 22: Range measurements of the different anchors vs. measured time. All UWB ranges are compared with their respective range from the reference system.

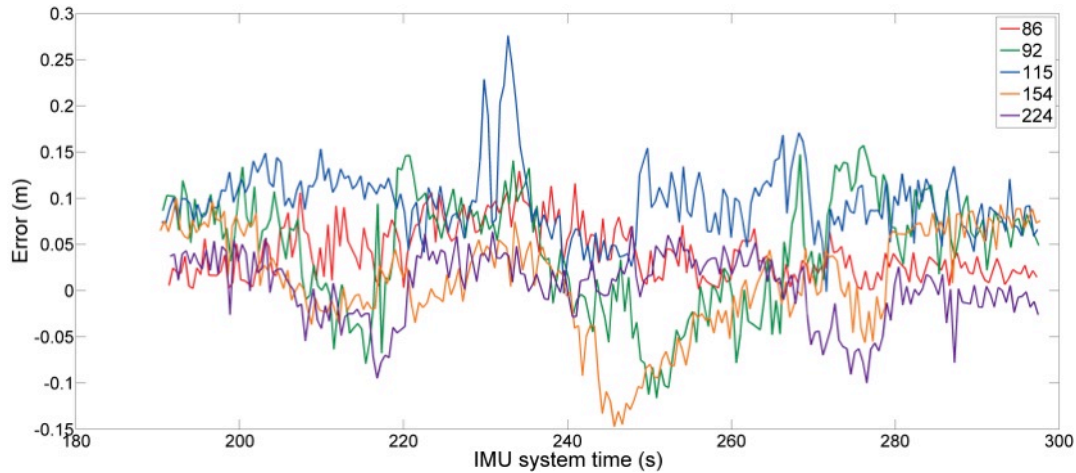


Figure 23: UWB range measurement errors calculated as the difference between UWB range value and reference system range value.

#### 4.1.4.2 Multilateration

In figure 24 the range data from figure 22 have been solved for position with multilateration, as outlined in the theory section. The position shows a larger error at the beginning and at the end of the measurement. The z-axis shows a much bigger error than the x- and the y-axis values. The normalized residual from minimising the cost position function is presented in figure 23 and is used to evaluate the performance of the filter and the post processing solution.

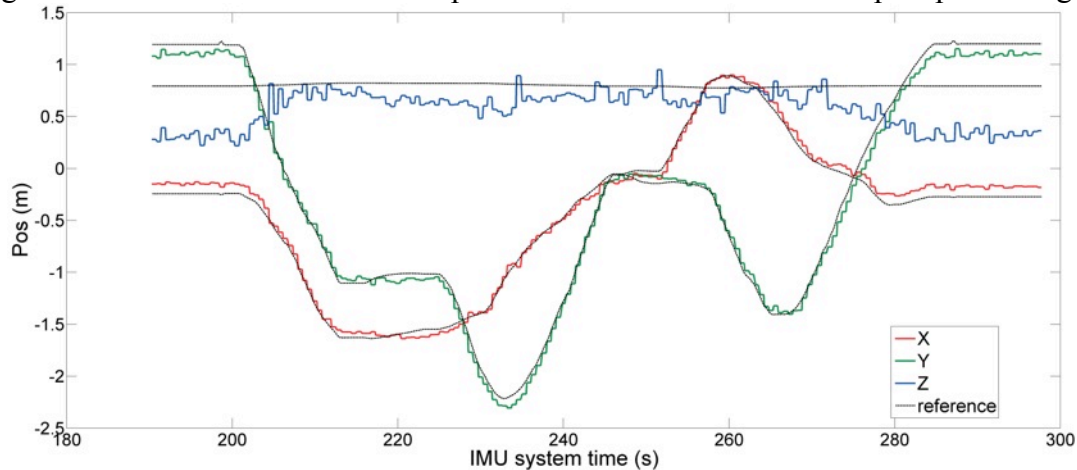


Figure 24: Position calculated from only UWB range data. The position is compared with the reference system position.

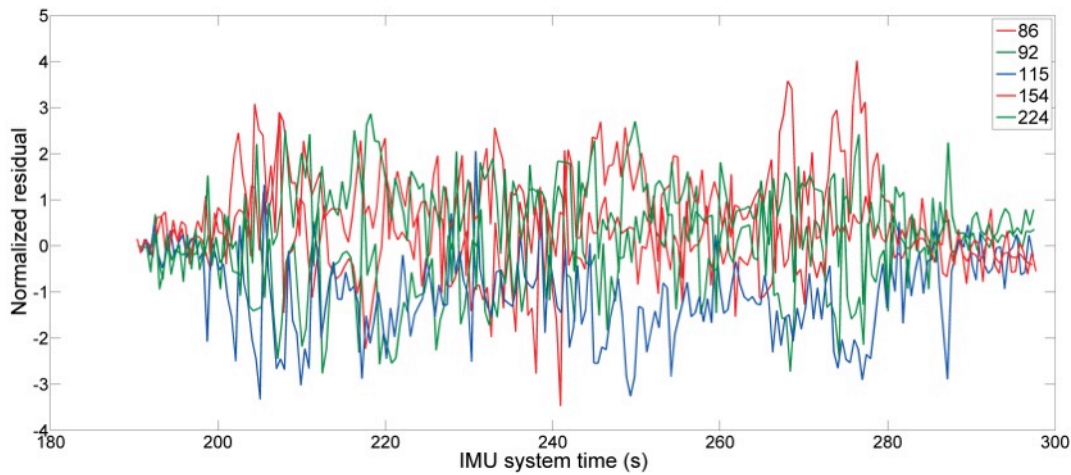


Figure 25: Normalized residual from the multilateration.

#### 4.1.4.3 Sensor fusion

The sensor fusion is evaluated after sensor fusion with the filter (Figures 26, 27 and 28) and after post processing of the filter solution (Figures 29, 30 and 31)

In figure 26 the range data from figure 22 have been solved for position with multilateration while fusing it with the IMU data from figure 20, using the XEE-filter outlined in the theory section. The position shows a larger error at the beginning and at the end of the measurement. The z-axis shows a much bigger error than the x- and the y-axis values. The normalized residual from the filter solution is presented in figure 27 and shows a slight increase compared to the multilateration residual in figure 25. Figure 28 shows x-position vs. y-position from the multilateration solution in figure 24 compared to the filter solution in figure 26 along with the reference system position.

In figure 29 the filter solution from figure 24 has been passed through an XEE post processing filter. In comparing with figure 24, the position still shows a larger error at the beginning and at the end of the measurement, with the z-axis showing the biggest error. The normalized residual from the filter solution is presented in figure 30 and shows roughly the same amplitudes as in the filtered solution from figure 27. Figure 31 shows x-position vs. y-position from the multilateration solution (from figure 24) compared to the post processed solution (from figure 29) along with the reference system position.

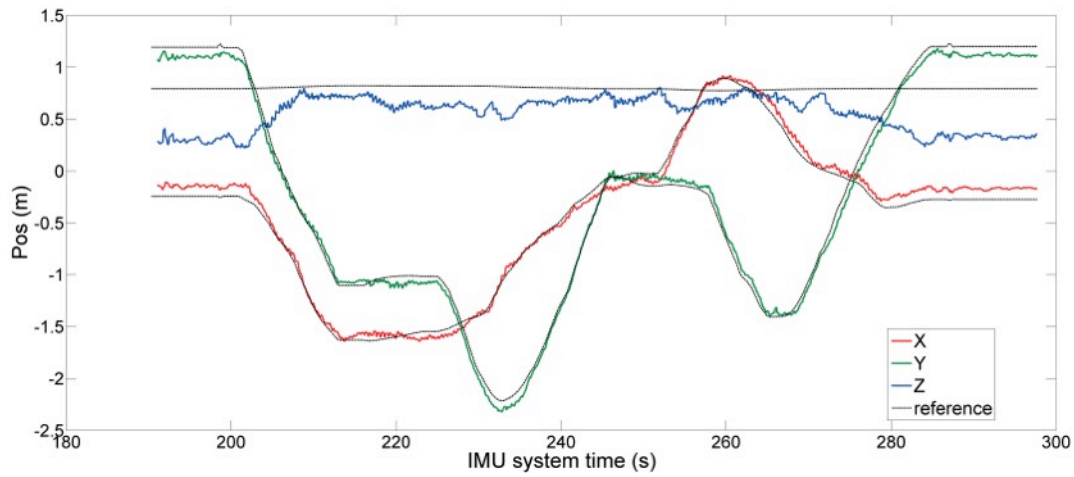


Figure 26: Position calculated from UWB range data fused with IMU accelerometer data. The position is compared with the reference system position.

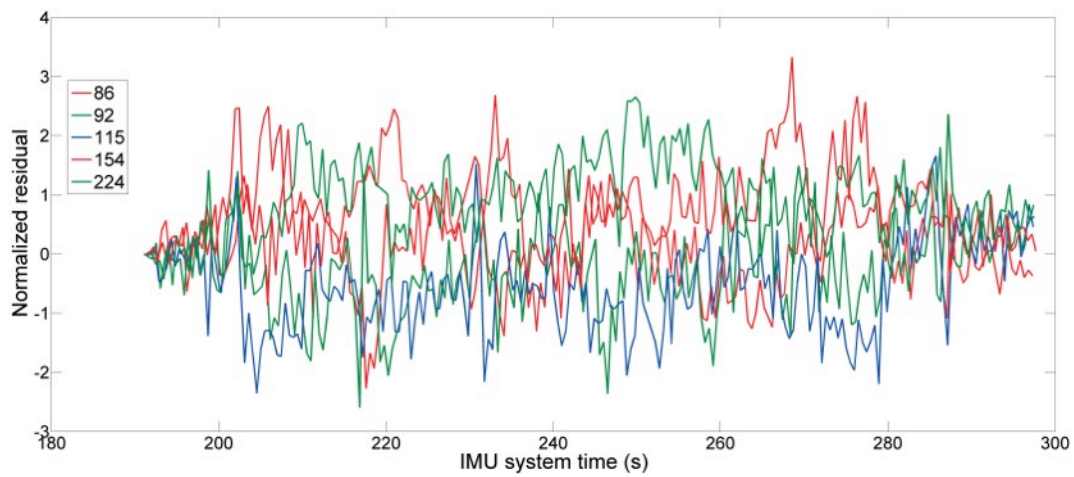


Figure 27: Normalized residual from the sensor fusion.



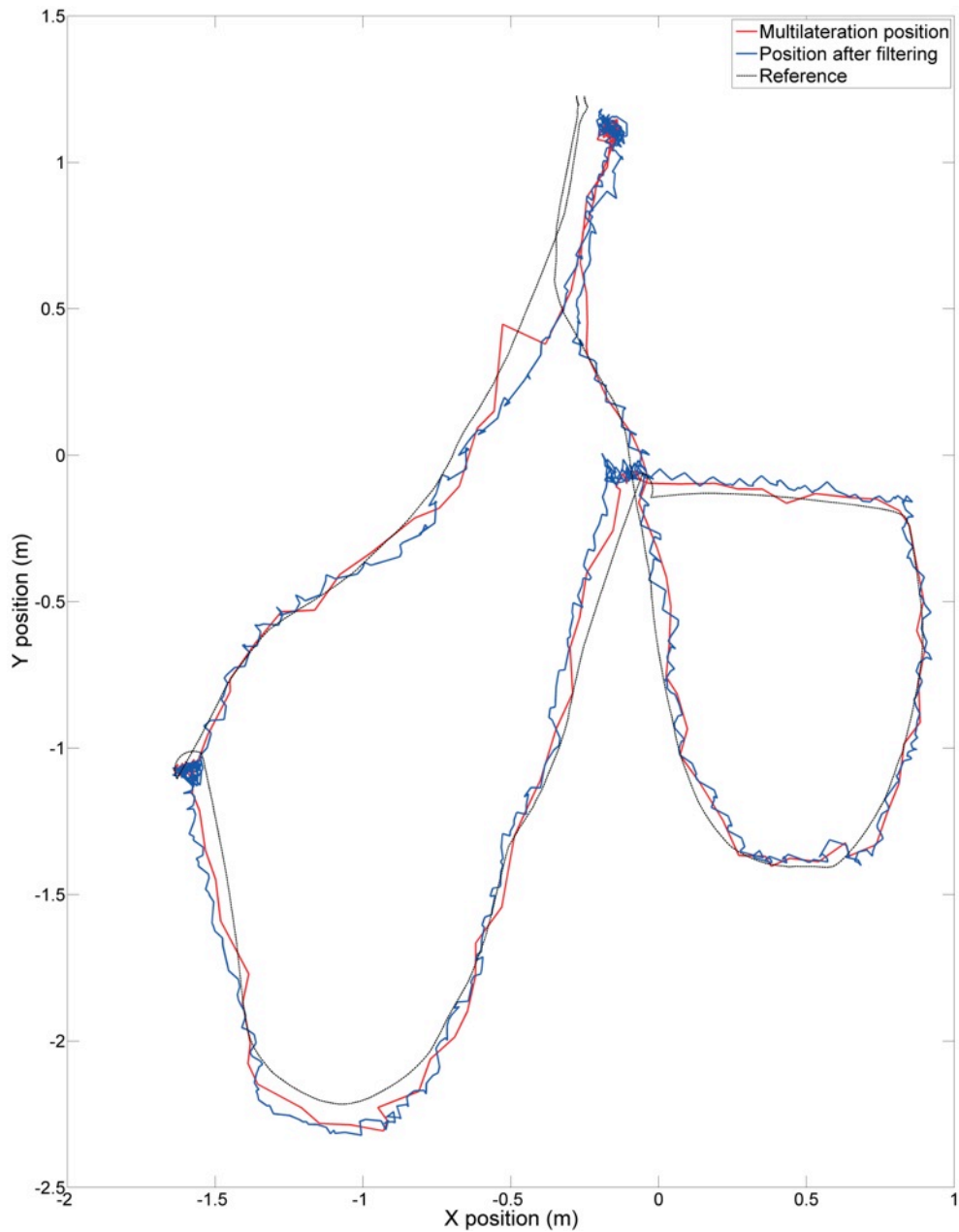


Figure 28: x- and y-position calculated from UWB range data fused with IMU accelerometer data. The position is compared with position from multilateration of UWB data and with the reference system position.

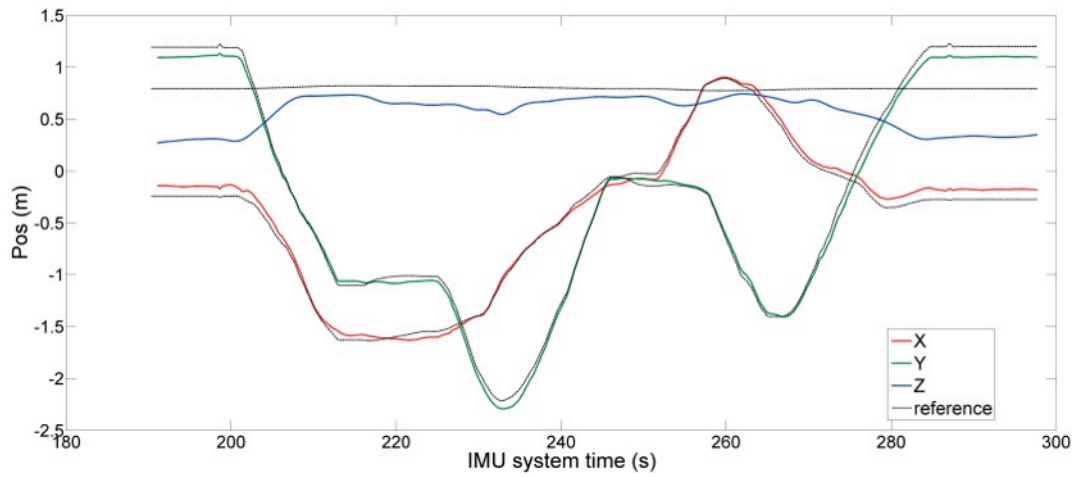


Figure 29: Position calculated from UWB range data fused with IMU accelerometer data. The final solution is achieved with post processing. The position is compared with the reference system position.

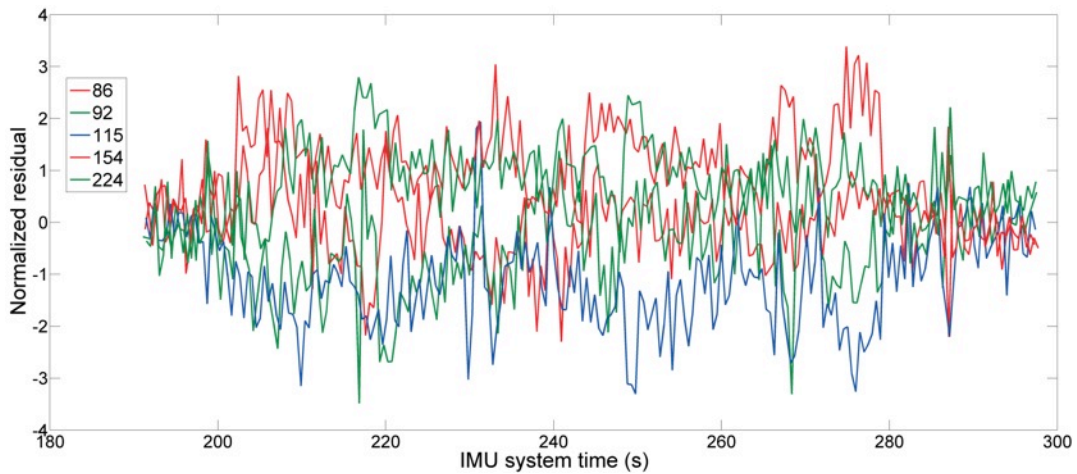


Figure 30: Normalized residual from the post processing.



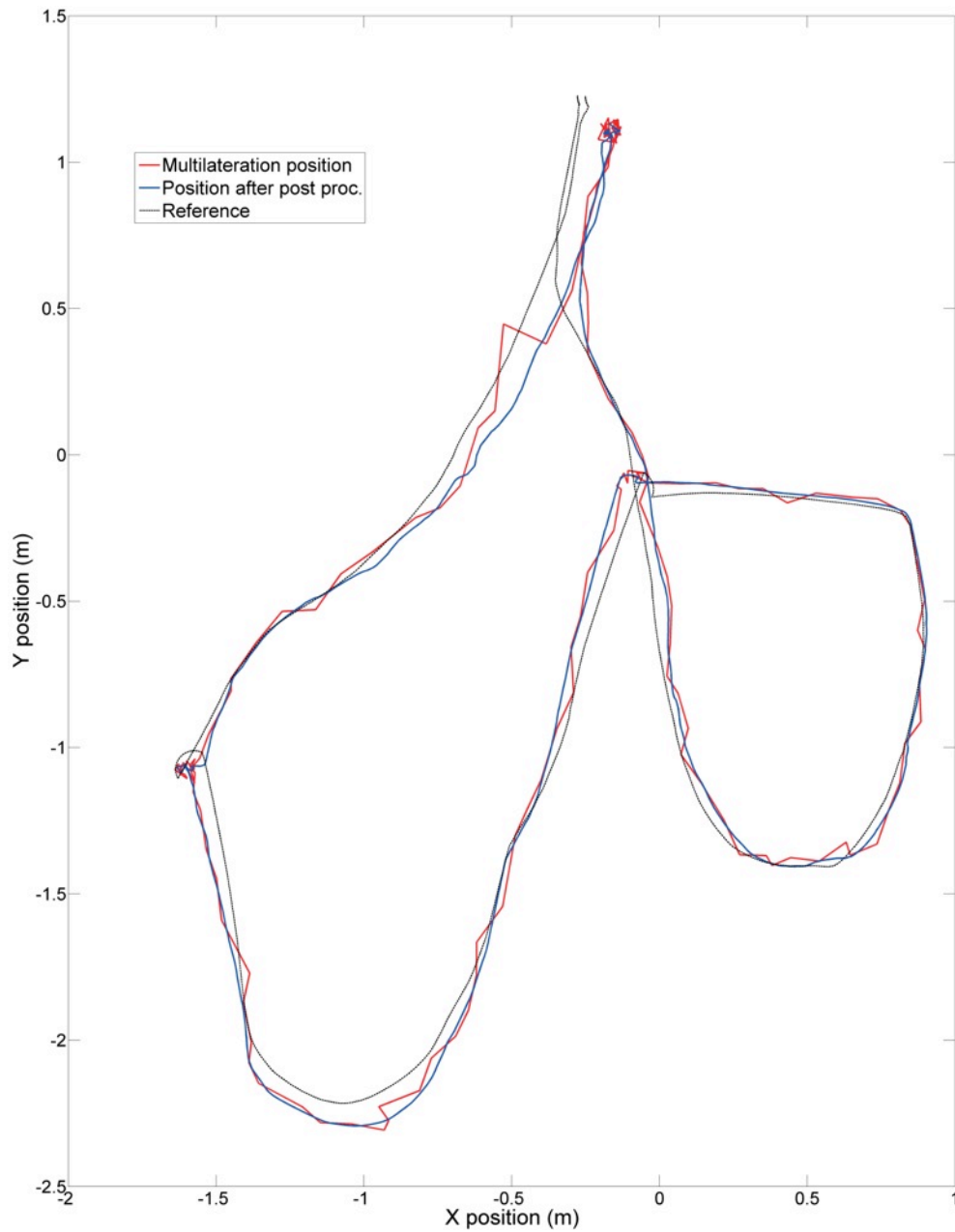


Figure 31: x- and y- position calculated from UWB range data fused with IMU accelerometer data. The final solution is achieved with post processing. The position is compared with position from multilateration of UWB data and with the reference system position

## 5 Conclusion and discussion

The long run measurement shows that the UWB ranging gives enough precision to be a viable option in the type of positioning required. It is obvious from figure 8 that the anchors perform differently under similar conditions as anchor 82 and anchor 96 shows a higher noise level than other anchors. This could indicate quality differences in the hardware. The error differences are further established in the histogram in figure 9, where anchor 82 and 96 deviates more from their mean values than the other anchors. Again in figure 10, tags 86 and 92 stand out with a higher random walk and a higher bias. The rotation measurement indicates that antenna direction does have some effect on the ranging precision. Comparisons of figure 11, 12 and 13 respectively show a slight correlation between range error and antenna direction. The general error is however not much larger than what can be seen in the long run measurement, presented e.g. in figure 8. The translation measurements gives some errors, similar to the ones that is being accounted for with the bias table, described in the theory section, and could possibly be because the bias table is not optimally calibrated.

Considering the long run measurement, the rotation measurement and the translation measurements the conclusion is that the UWB would be well suited for the positioning demands posed by the robot platform. Further tests are recommended to determine antenna behaviour.

For the final sensor fusion solution it is clear that the two sensors combined performs better than either of the sensors running alone. Some improvement can be seen when comparing the multilateration positioning with the sensor fusion solution in figure 28 and with the post processed filter solution in figure 31. The relatively lower resolution of the multilateration positioning is because the algorithm uses five range values for every position point, whereas the sensor fusion solution uses separate UWB range values and calculates position between with the help of IMU data. There are some large errors in the z-axis positioning, clearly visible in figure 24. These are most likely because the anchor spacing was lower along z-axis compared to the x- and y-axis. For the purpose of planar motion however, the z-axis precision is less important than the x- and y-axis precision. Figures 22 and 23 show some large unexplained errors. Most likely these are because none-line-of-sight conditions, but without isolated tests of none-line-of-sight conditions this cannot be fully established. Before post processing of the sensor fusion solution some choppy behaviour can be seen in figures 26 and 28. This is possibly because of drifting from integrating the accelerometer data from the IMU.

The conclusion from the tests of the sensor fusion solution is that positioning works well when using a sensor fusion of UWB and IMU sensors. The difference between multilateration solution and sensor fusion solution would possibly be larger with a higher movement speed of the sensor platform, since the sample rate of the UWB is much slower than that of the IMU. Further tests with different movement speeds are therefore recommended, as well as further tests under none-line-of-sight conditions with different type of materials are recommended.

## **5.1 Error sources**

The largest sources of error that has been discussed so far is most likely the low differences in anchor node z-positions. When performing multilateration, the anchor nodes should be placed as far away from each other as possible in all dimensions in order to minimize the magnification of positioning error from ranging measurement noise [3]. Because of limitations in the measurement area, the difference in height, or z-position, between the anchor nodes is smaller than the difference in x and y position. As a result, the error of the z-position is the largest as can be seen in figure 24, 26 and 29. Also discussed were the errors that come from precision differences between different anchor nodes. Both the anchor node positions and precisions are fairly easy to compensate for and would in doing so likely lead to a lower positioning error. An error that is less trivial to compensate for lies in the orientation of the antennas on the UWB boards. From the rotation measurement in figure 11, the error seems to correlate with antenna rotation. This needs to be further examined.

All errors are calculated with respect to the visual reference system. The reference system uses markers that are placed on the antennas of the UWB boards. Because of the size of the markers, their positions are offset by 1-2 cm to the antennas. This is not adjusted for when calculating the errors and therefore affects the result.

## **5.2 Suggestions for further development**

The positioning is currently done by processing log files in Matlab. In order to make the sensor platform a feasible solution for real time positioning, further development should include making the positioning real time and less computationally heavy.

Measurements are started by running the IMU and the two-way ranging programs separately, since for unknown reasons running them in one program can cause the Raspberry Pi to slow down to the point when errors occur in the ranging. Investigating and solving this issue would increase the robustness of the measurements.

In an implementation on a robot platform with motor control data available, better positioning could possibly be obtained by adding this data to the sensor data in the sensor fusion.

There are several improvements to be made to increase the frequency of the UWB ranging where the largest limiting factor currently is the sleep time of 400 ms each anchor does after a ranging. By reprogramming the code on the ARM processor, the anchors could still be used without a host system but allow for much faster ranging. Another limiting factor is the speed of the SPI communication between the Raspberry Pi and the UWB board. This speed has a maximum limit of 20 MHz on the UWB board, but could not reliably be set to any higher than 7 Mhz. A full list of issues can be found in the appendix (7.1)



## 6 References

[1] O. J. Woodman. *An introduction to inertial navigation*. Technical Report UCAM- CL-TR-696, University of Cambridge, Computer Laboratory, Aug. 2007.

[2] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer-Verlag, New York, 2nd edition, 2006. ISBN 0387987932.

[3] T. Kunz and B. Tatham. *Localization in Wireless Sensor Networks and Anchor Placement*. Journal of Sensor and Actuator Networks, June 2012

[4] DecaWave Ltd. *DW1000 User Manual*. <http://www.decawave.com/>

J. Hol. *Sensor Fusion and Calibration of Inertial Sensors, Vision, Ultra-Wideband and GPS*. Linköping University, Department of Electrical Engineering, 2011

## 7 Appendix

### 7.1 Known issues

- The major bottleneck in the UWB sampling rate is the pre-determined sleep time of 400ms for the anchor nodes after a successful ranging. It is possible that this sleep time can be set from the Decaranging software provided with the evaluation set, or else by programming directly to the on board ARM processor on the EVB1000.
- The anchors can stop blinking spontaneously when starting up for ranging, meaning that the ranging is never initiated. Hard resetting the stopped tags helps.
- The DW1000 ARM processor has a built in feature that on some errors halts the system until the system clock resets. A full system clock revolution (the system time being a 40 bit register) is roughly 17.2 s and there is therefore a risk that an anchor will stop ranging for this duration or less. This issue is considered rare.
- There is a chance of the tag node reading old time values (transceiver time stamp) from its register which gives large negative ranging values. This is currently fixed in post processing by removing faulty values and their corresponding sync bits in the IMU data.
- Currently the SPI bus communication is unable to reach its potential CLK speed setting. The UWB SPI pins are rated at 20 MHz and the Raspberry Pi SPI pins are rated at 125 MHz. Currently, the top CLK setting that works is 7MHz before the signal deteriorates. Wires are twisted to counteract antenna behavior interference.
- Tests with running the two sensors from the same script results in delays in the ranging, giving a “transceiver startup power error” pausing the script and failing the current ranging. This also leads to the number of received sync pulses on the IMU to differ from the number of sync pulses sent from the UWB. The current solution is to run the two sensor scripts separately on separate SSH sessions.
- It is currently possible to exit ranging between receiving a final message and logging ranging data. On these occasions the IMU will receive one sync pulse more than there are UWB ranges. This is handled by removing last received IMU sync flag if the IMU sync flags numbers one more than logged UWB ranges.
- The CLI output from the UWB ranging script sometimes freezes for short periods. This is however not noted in the UWB log file system times and is

therefore most likely a result of poor Wi-Fi connection.

- Ranging with anchor 92 sometimes detects ranges that differ from real value with approx. 5 meters. This is considered rare.
- Scanning ports when initiating IMU measurements sometimes misses the connected IMU device. This is considered common.
- Running the IMU script several times sometimes slows down the system to a point where the two programs does not work as intended. This includes missed messages and not ranging with all tags. This is currently solved by rebooting the Raspberry Pi. This is considered common.
- The UWB ranging sometimes will not initiate. This is solved by restarting the UWB script. This is considered common.
- A GLIBC error is sometimes detected upon exiting the IMU script or the UWB script.

## 7.2 UWB message format

### 7.2.1 IEEE 802.15.4 UWB physical layer

The DW1000 transceiver IC implements the UWB physical layer as specified in the IEEE 802.15.4 - 2011 standard.

Preamble	Start of Frame Delimiter	PHY header	Data
64, 1024 or 1096 symbols	8 or 64 symbols	19 bits	Up to 127 coded octets

Figure 32: UWB physical layer frame structure

There are several configuration options available for the standard. For more details regarding these, the reader is referred to the DW1000 user manual [4]. After the initial configuration of the DW1000 at startup, the IC handles every part but the MAC part of the frame.

### 7.2.2 IEEE 802.15.4 MAC layer

In the data part of UWB frame shown in figure 32, the MAC message is stored. This part can according to the standard be of variable length up to 127 octets, or up to 1023 octets using the DW1000's non-standard long frame mode option. Implementation of the MAC layer is the responsibility of the host system since the DW1000's transceiver does not include all the functionality needed for this.

Mac Header							Mac Payload	Mac Footer
Frame control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Source Address	Aux Security Header	Frame Payload	FCS
2 octets	1 octet	0 or 2 octets	0, 2 or 8 octets	0 or 2 octets	0, 2 or 8 octets	5, 6, 10 or 14 octets	Variable number of octets	2 octets

Figure 33: General MAC message format

The MAC layer consists of the MAC Header, Payload and Footer as seen in figure 33. The DW1000 includes a frame filtering algorithm, which when turned on parses the header to determine if the frame should be accepted or not by checking if the destination address matches the address of the address information programmed in the IC or if the frame is a broadcast message. The only part not needed by the host system to implement is the MAC footer, which is inserted by the DW1000 as the last two octets of the payload.

## 7.3 Raspberry Pi setup

### 7.3.1 General setup

- The Wi-Fi access point is using WPA encryption. SSID is set to “raspi” and password is set to “raspberry”. Note that the access point does not utilize DHCP and thus all devices connecting to the network needs a static IP address.
- The Raspberry pi is running the Debian-derived operating system Raspian. Functionality for SSH and Wi-Fi is enabled from start. The packages used are the latest update from April 20, 2014.
- All software for sensor data acquisition is compiled on the Raspberry Pi.

### 7.3.2 Wi-Fi setup

- To configure a setup “home” to use the desired network (see *General setup*) the following needs to be added to the file `/etc/wpa_supplicant/wpa_supplicant.conf`:

```
network={
    ssid="raspi"
    scan_ssid=1
    psk="raspberry"
    key_mgmt=WPA-PSK
    id_str="home"
    priority=15
}
```

- To enable the network “home” settings on Wi-Fi, the following needs to be added to the file `/etc/network/interfaces`:

```
iface home inet static
    address 192.168.1.1
    netmask 255.255.255.0
```



Note that the interfaces setup still enables DHCP, making it possible to connect to DHCP networks via Ethernet. For temporarily disabling DHCP usage, while running Wi-Fi, the file `/etc/resolv.conf` has had every line commented out. This is automatically changed back on connection via Ethernet as a standard setting in `/etc/network/interfaces`.

### 7.3.3 Preparing for the UWB sensor

- In order to use the GPIO pins for SPI communication the linux kernel needs the added module `spi_bcm2708`. This is achieved with a script `loadSpiModule` saved in the user root (`~/`) that loads the module and allows user access. This contains the following:

```
sudo modprobe spi_bcm2708
sudo chown `id -u`.`id -g` /dev/spidev0.*
```

- To enable the kernel module to load on boot up the following needs to be added to the file `/etc/rc.local`:

```
su pi -c '/bin/bash ~/loadSpiModule'
```

- The GPIO access library `WiringPi`, available at <http://wiringpi.com/>, is used to handle UWB IRQ via the Raspberry Pi GPIO pins.

### 7.3.4 Preparing for the IMU sensor

- For IMU functionality the Xsens SDK needs to be installed. It is recommended that a build is made on the Raspberry Pi to ensure that shared objects and static libraries are compatible. The GCC compiler needs to be 4.5 or higher. The version used (see *General setup*) is version 4.6.3.